REMARKS

Claims 1-68 remain pending in the application. Reconsideration is respectfully requested in light of the following remarks.

Objection to the Drawings:

The Examiner objected to Fig. 10b in regard to reference numeral 130 which is referred to as 180 in the specification. In light of the drawing amendment made herein, Applicants respectfully request withdrawal of the objection to the drawings.

Section 102(e) Rejection:

The Office Action rejected claims 1-8, 10-17, 22-31, 33-37, 42-46, 48-52, 54, 59-65 and 68 under 35 U.S.C. § 102(e) as being anticipated by Bittenger et al. (U.S. Patent 6,453,362) (hereinafter "Bittenger").

Regarding claim 1, the Examiner states that Bittenger discloses "remotely invoking methods in a distributed computing environment, comprising: A client generating a message in a data representation language, wherein the message includes information representing a computer programming language method call, and wherein the message further includes a credential for allowing the client access to a service configured to perform functions on behalf of clients in the distributed computing environment." Applicants respectfully disagree with the Examiner's interpretation of Bittenger.

Bittenger teaches a method wherein a client invokes a remote server application using a server stub object that packs or marshals method parameters into a request message (Bittenger, column 6, lines 9-17, and lines 28-33). Bittenger also discloses the use of systems such as CORBA, DCOM, RMI and Java to invoke methods on remote server applications. However, Applicants can find no teaching in Bittenger regarding

generating a message in a data representation language. Note that CORBA, DCOM, RMI and Java are all code-centric technologies that do not employ data representation language messages for method calls.

Additionally, Bittenger teaches the use of a ticket object created by the client that a server application can use to pass a server stub object to the client. The client can then use the server stub object to invoke remote methods on the server application (Bittenger, column 6, lines 63-67, column 7, lines 42-47, and column 8, lines 15-22). Hence, under Bittenger, the ticket is used to notify the client that the server application is running and ready to receive requests and provides a method for the server application to provide a server stub object to the client. Further, Bittenger teaches the user of a separate authentication server to authenticate the client using well-known login procedures such as user id and password. After authenticating the client, the authentication server then forwards the ticket to the server application (Bittenger, column 7, line 67 – column 8, line 8). Therefore Applicants assert that Bittenger does not teach the use of a ticket as a credential as suggested by the Examiner.

Thus, Applicants submit that Bittenger fails to disclose a method for remotely invoking methods in a distributed computing environment, comprising: A client generating a message in a data representation language, wherein the message includes information representing a computer programming language method call, and wherein the message further includes a credential for allowing the client access to a service configured to perform functions on behalf of clients in the distributed computing environment.

Further regarding claim 1, the Examiner states, "Bittenger discloses a method for remotely invoking methods in a distributed computing environment, comprising ... [t]he service examining the credential included in the message." Applicants respectfully disagree with the Examiner's interpretation of Bittenger.

As discussed above, Bittenger fails to teach the inclusion of a credential with a remote method request. Hence, Bittenger also fails to teach the service examining such a credential. In contrast, Bittenger teaches the use of a server authenticating a client prior to the launching of the desired server application (Bittenger, column 8, lines 9-14). Bittenger fails to teach the service application itself verifying any credentials from the client. Thus, applicants submit that Bittenger does not disclose a method comprising a service examining a credential included in the message.

In light of the above remarks, Applicants assert that the rejection of claim 1 is not supported by the cited art and withdrawal of the rejection is respectfully requested.

Regarding claim 5, the Examiner states that Bittenger discloses "the client message endpoint attaching the credential to the message (tStamp is an identifier used on all messages)." Applicant disagrees with the Examiner.

Bittenger teaches that "the identifier 'tstamp' may be used to locate the ticket within the [client-side registry] database" (Bittenger, column 7, lines 8-9). Bittenger further describes a server using the tstamp to retrieve a ticket stub from the client-side registry (Bittenger, column 7, lines 41-43). Applicant can find no teaching in Bittenger that a tstamp is used on all messages as the Examiner contends. Further, applicants assert that Bittenger's use of a tstamp to allow a server application to locate and retrieve a ticket stub from a client-side registry does not constitute a client message endpoint attaching the credential to the message.

In light of the above remarks, Applicants assert that the rejection of claim 5 is not supported by the cited art and withdrawal of the rejection is respectfully requested.

Regarding claim 6, the Examiner states that Bittenger discloses "the service providing to the client a service advertisement comprising a data representation language message schema comprising descriptions of data representation language messages the client is authorized to send to the service, and wherein said generation a message is

performed in accordance with a description of the message comprises in the message schema." The Examiner supports this erroneous contention by claiming "the server stub [as taught by Bittenger] is sent to the client as a parameter to define messages and request the client can make." Applicant disagrees with the Examiner's interpretation of Bittenger.

Bittenger clearly describes that the server stub is a part of the use of RMI for invoking a remote server application. (Bittenger, column 5, line 55 – column 6, line 41, and specifically, column 6, lines 28-38). Applicants assert that RMI does not involve a service providing to the client a service advertisement comprising a data representation language message schema comprising descriptions of data representation language messages the client is authorized to send to the service.

Further, Applicants can find no mention at all in Bittenger regarding service advertisements or data representation schemas, and specifically can find no suggestion of using a data representation schema in a service advertisement.

Additionally, Applicants assert that the server stub as taught be Bittenger cannot represent a service advertisement as the Examiner's contends because Bittenger teaches that the server application sends the server stub to the client only after the client has already requested that the server applicant be started and using a ticket stub which the server application obtained from the client-side registry using a ticket identifier supplied by the client application (Bittenger, column 8, lines 6-24). Clearly the server stub cannot be a service advertisement if the client has already initiated communications with the desired server application.

As such, Applicants assert that the rejection of claim 6 is not supported by the cited art and withdrawal of the rejection is respectfully requested. Similar remarks as discussed above in regard to claim 6 apply to claims 29 and 60.

Regarding claim 7, the Examiner asserts that "Bittenger discloses the client generating a client method gate in accordance with the service advertisement, wherein the client method gate is configured to provide to the client an interface to the service by generating the data representation language messages described in the message schema, wherein said generating a message is performed by the client method gate" and further supports this assertion by claiming "the server stub [as taught by Bittenger] is sent to the client as a parameter to define messages and requests the client can make." Applicants disagree with the Examiner's interpretation of Bittenger.

As stated above, Bittenger fails to teach a service advertisement and also fails to teach a message schema describing data representation language messages. Applicant submits that Bittenger necessarily must also fail to teach a client generating a client method gate in accordance with a service advertisement and further fails to teach the client method gate providing the client an interface to the service by generating data representation language messages described in the message schema.

Therefore, Applicants assert that the rejection of claim 7 is not supported by the cited art and withdrawal of the rejection is respectfully requested. Similar remarks as discussed above in regard to claim 7 apply to claims 31 and 61.

Regarding claim 8, the Examiner contends that "Bittenger discloses the service advertisement further comprises an address for receiving the data representation language messages on the service, wherein said sending the message to the service comprises sending the message to the address." Applicants respectfully disagree with the Examiner's interpretation of Bittenger.

Applicants assert that Bittenger fails to disclose a service advertisement comprising an address for receiving data representation language messages on the service. As shown above, Bittenger clearly fails to disclose either a service advertisement or data representation language messages. Clearly, if Bittenger does not disclose either a service advertisement or data representation language messages,

Bittenger cannot disclose a service advertisement that comprises an address for receiving data representation language messages on a service.

Therefore, Applicants assert that the rejection of claim 8 is not supported by the cited art and withdrawal of the rejection is respectfully requested. Similar remarks as discussed above in regard to claim 8 apply to claim 31.

Regarding claim 10, the Examiner claims that "Bittenger discloses the client generating a client message endpoint in accordance with the service advertisement..." and supports this claim by stating "the server stub, originally sent to the client, is re-generated by the client to act as an endpoint." Applicants disagree with the Examiner's interpretation of Bittenger.

As shown above Bittenger fails to teach the use of a service advertisement. Additionally, at the Examiner's cited passes, Bittenger teaches that "the ticket saves the server stub so that the server stub is accessible to the client application ..." and that the "client application can then user the server stub to make requests to the application" (Bittenger, column 7, lines 50-57). Applicants can find no reference in Bittenger referring to a client *re-generating* the server stub as the Examiner has stated.

Therefore, Applicants assert that the rejection of claim 10 is not supported by the cited art and withdrawal of the rejection is respectfully requested.

Regarding claim 25, the Examiner states, "Bittenger discloses a distributed computing system comprising: ... client device configure to: generate a message in a data representation language, wherein the message includes information representing a computer programming language method call, and wherein the message further includes a credential for allowing the client access to the service device." Applicants respectfully disagree with the Examiner's interpretation of Bittenger.

Bittenger teaches a method wherein a client invokes a remote server application using a server stub object that packs or marshals method parameters into request message (Bittenger, column 6, lines 9-17, and lines 28-33). Bittenger also discloses the use of systems such as CORBA, DCOM, RMI and Java to invoke methods on remote server applications. Applicants can find no teaching in Bittenger regarding the use of a data representation language.

Additionally, Bittenger teaches the use of a ticket object created by the client that a server application can use to pass a server stub object. The client can then use the server stub object to invoke remote methods on the server application (Bittenger, column 6, lines 63-67, column 7, lines 42-47, and column 8, lines 15-22). Hence, under Bittenger, the ticket is used to notify the client that the server application is running and ready to receive requests and provides a method for the server application to provide a server stub object to the client. Further, Bittenger teaches the user of a separate authentication server to authenticate the client using well-known login procedures such as user id and password. After authenticating the client, the authentication server then forwards the ticket to the server application (Bittenger, column 7, line 67 – column 8, line 8). Therefore Applicants assert that Bittenger does not teach the creating and using of a ticket as a credential as suggested by the Examiner.

Thus, Applicants submit that Bittenger fails to disclose system comprising a client device configure to generate a message in a data representation language, wherein the message includes information representing a computer programming language method call, and wherein the message further includes a credential for allowing the client access to the service device.

Further regarding claim 25, the Examiner states, "Bittenger discloses a distributed computing system ...[w]herein the service device is configured to examine the credential included in the message." Applicants respectfully disagree with the Examiner's interpretation of Bittenger.

As discussed above, Bittenger fails to teach the inclusion of a credential with a remote method request. Hence, Bittenger also fails to teach the service device examining such a credential. Bittenger teaches the use of an authentication server authenticating a client through standard authentication techniques, such as rlogin prior to the launching of the desired server application (Bittenger, column 7, lines 15-17 and column 8, lines 9-14). Bittenger fails to teach the server application itself verifying any credentials from the client. Thus, applicants submit that Bittenger does not disclose a system wherein a service is configured to examine a credential included in the message.

In light of the above remarks, Applicants assert that the rejection of claim 25 is not supported by the cited art and withdrawal of the rejection is respectfully requested.

Regarding claim 45, the Examiner states, "Bittenger discloses a device comprising ... a method gate...wherein the method gate in configured to: [g]enerate a message in a data representation language, wherein the message includes information representing a computer programming language method call, and wherein the message further includes a credential for allowing the client access to a service ...." Applicants respectfully disagree with the Examiner's interpretation of Bittenger.

Bittenger teaches a method wherein a client invokes a remote server application using a server stub object that packs or marshals method parameters into request message (Bittenger, column 6, lines 9-17, and lines 28-33). Bittenger also discloses the use of systems such as CORBA, DCOM, RMI and Java to invoke methods on remote server applications. Applicants can find no teaching in Bittenger regarding the use of a data representation language.

Additionally, Bittenger teaches the use of a ticket object created by the client that a server application can use to pass a server stub object. The client can then use the server stub object to invoke remote methods on the server application (Bittenger, column 6, lines 63-67, column 7, lines 42-47, and column 8, lines 15-22). Hence, under Bittenger, the ticket is used to notify the client that the server application is running and

ready to receive requests and provides a method for the server application to provide a server stub object to the client. Further, Bittenger teaches the user of a separate authentication server to authenticate the client using well-known login procedures such as user id and password. After authenticating the client, the authentication server then forwards the ticket to the server application (Bittenger, column 7, line 67 – column 8, line 8). Therefore Applicants assert that Bittenger does not teach the creating and using of a ticket as a credential as suggested by the Examiner.

Thus, Applicants submit that Bittenger fails to disclose a device comprising a method gate wherein the method gate in configured to generate a message in a data representation language, wherein the message includes information representing a computer programming language method call, and wherein the message further includes a credential for allowing the client access to a service.

Further regarding claim 45, the Examiner states, "Bittenger discloses a device ...[w]herein the service is operable to verify the message as authentic by examining the credential included in the message." Applicants respectfully disagree with the Examiner's interpretation of Bittenger.

As discussed above, Bittenger fails to teach the inclusion of a credential with a remote method request. Hence, Bittenger also fails to teach the service examining such a credential. Bittenger teaches the use of an authentication server authenticating a client through standard authentication techniques, such as rlogin prior to the launching of the desired server application (Bittenger, column 7, lines 15-17 and column 8, lines 9-14). Bittenger fails to teach the server application itself verifying any credentials from the client. Applicants therefore submit that Bittenger discloses authenticating the client, not a message. Thus, applicants submit that Bittenger does not disclose a system wherein a service is operable to verify a message as authentic by examining a credential included in the message.

In light of the above remarks, Applicants assert that the rejection of claim 45 is not supported by the cited art and withdrawal of the rejection is respectfully requested.

Regarding claim 49, the Examiner states, "Bittenger discloses a device comprising: [a] client component configured to generate a message in a data representation language, wherein the message includes information representing a computer programming language method call." Applicants respectfully disagree with the Examiner's interpretation of Bittenger.

Bittenger teaches a method wherein a client invokes a remote server application using a server stub object that packs or marshals method parameters into request message (Bittenger, column 6, lines 9-17, and lines 28-33). Bittenger also discloses the use of systems such as CORBA, DCOM, RMI and Java to invoke methods on remote server applications. Applicants can find no teaching in Bittenger regarding the use of a data representation language. Thus, Applicants submit that Bittenger fails to disclose a device comprising a client component configured to generate a message in a data representation language, wherein the message includes information representing a computer programming language method call.

Further regarding claim 49, the Examiner states, "Bittenger discloses a device comprising ... [a] message endpoint configured to: [a]ttach a credential to the message for allowing the client component access to a service in a distributed computing environment." Applicants disagree with the Examiner's interpretation of Bittenger.

Bittenger teaches the user of a separate authentication server to authenticate the client using well-known login procedures such as user id and password. After authenticating the client, the authentication server then forwards the ticket to the server application (Bittenger, column 7, line 67 – column 8, line 8). Further, Bittenger teaches the use of a ticket object created by the client that a server application can use to pass a server stub object. The client can then use the server stub object to invoke remote methods on the server application (Bittenger, column 6, lines 63-67, column 7, lines 42-

47, and column 8, lines 15-22). Hence, under Bittenger, the ticket is used to notify the client that the server application is running and ready to receive requests and provides a method for the server application to provide a server stub object to the client. Therefore, Applicants submit that Bittenger does not teach creating and using a ticket as a credential as suggested by the Examiner.

Thus, Applicants submit that Bittenger fails to disclose a device comprising a message endpoint configured to attach a credential to the message for allowing the client component access to a service in a distributed computing environment.

Further regarding claim 49, the Examiner states, "Bittenger discloses a device ...[w]herein the service is operable to verify the message as authentic by examining the credential included in the message..." Applicants respectfully disagree with the Examiner's interpretation of Bittenger.

As discussed above, Bittenger fails to teach the inclusion of a credential with a remote method request. Hence, Bittenger also fails to teach the service examining such a credential. Bittenger teaches the use of an authentication server authenticating a client through standard authentication techniques, such as rlogin prior to the launching of the desired server application (Bittenger, column 7, lines 15-17 and column 8, lines 9-14). Bittenger fails to teach the server application itself verifying any credentials from the client. Applicants therefore submit that Bittenger discloses authenticating the client, not a message. Thus, applicants submit that Bittenger does not disclose a device wherein the service is operable to verify the message as authentic by examining the credential included in the message.

In light of the above remarks, Applicants assert that the rejection of claim 49 is not supported by the cited art and withdrawal of the rejection is respectfully requested.

Regarding claim 59, the Examiner states, "Bittenger discloses a carrier medium comprising program instructions, wherein the program instructions are computer-

executable to implement: [a] client generating a message in a data representation language, wherein the message includes information representing a computer programming language method call, and wherein the message further includes a credential for allowing the client access to a service configured to perform functions on behalf of clients in the distributed computing environment." Applicants respectfully disagree with the Examiner's interpretation of Bittenger.

Bittenger teaches a method wherein a client invokes a remote server application using a server stub object that packs or marshals method parameters into request message (Bittenger, column 6, lines 9-17, and lines 28-33). Bittenger also discloses the use of systems such as CORBA, DCOM, RMI and Java to invoke methods on remote server applications. Applicants can find no teaching in Bittenger regarding the use of a data representation language.

Additionally, Bittenger teaches the use of a ticket object created by the client that a server application can use to pass a server stub object. The client can then use the server stub object to invoke remote methods on the server application (Bittenger, column 6, lines 63-67, column 7, lines 42-47, and column 8, lines 15-22). Hence, under Bittenger, the ticket is used to notify the client that the server application is running and ready to receive requests and provides a method for the server application to provide a server stub object to the client. Further, Bittenger teaches the user of a separate authentication server to authenticate the client using well-known login procedures such as user id and password. After authenticating the client, the authentication server then forwards the ticket to the server application (Bittenger, column 7, line 67 – column 8, line 8). Therefore Applicants assert that Bittenger does not teach the creating and using of a ticket as a credential as suggested by the Examiner.

Thus, Applicants submit that Bittenger fails to disclose a carrier medium comprising program instructions, wherein the program instructions are computer-executable to implement a client generating a message in a data representation language, wherein the message includes information representing a computer programming

language method call, and wherein the message further includes a credential for allowing the client access to a service configured to perform functions on behalf of clients in the distributed computing environment.

Further regarding claim 59, the Examiner states, "Bittenger discloses a carrier medium comprising program instructions, wherein the program instructions are computer-executable to implement ... [t]he service examining the credential included in the message." Applicants respectfully disagree with the Examiner's interpretation of Bittenger.

As discussed above, Bittenger fails to teach the inclusion of a credential with a remote method request. Hence, Bittenger also fails to teach the service examining such a credential. Bittenger teaches the use of an authentication server authenticating a client prior to the launching of the desired server application (Bittenger, column 8, lines 9-14). Bittenger fails to teach the server application itself verifying any credentials from the client. Thus, applicants submit that Bittenger does not disclose a carrier medium comprising program instructions, wherein the program instructions are computer-executable to implement a service examining a credential included in the message.

In light of the above remarks, Applicants assert that the rejection of claim 59 is not supported by the cited art and withdrawal of the rejection is respectfully requested.

Section 103(a) Rejections:

The Office Action rejected claims 9, 18-20, 32, 38-40, 47, 53, 55-58 and 66 under 35 U.S.C. § 103(a) as being unpatentable over Bittenger in view of Leach, et al. (U.S. Patent 6,108,715) (hereinafter "Leach"). Claims 21, 41 and 67 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Bittenger in view of Instaweb Online Computing Dictionary (Instaweb, http://www.instantweb.com/foldoc/foldoc.cgi?query=XML) (hereinafter "Instaweb").

Regarding claim 55, the Examiner states, "Bittenger discloses an endpoint configured to receive a message and verify the message as authentic..." Applicants disagree with the Examiner's interpretation of Bittenger.

As discussed above, Bittenger teaches the use of an authentication server authenticating a client through standard authentication techniques, such as rlogin prior to the launching a desired server application (Bittenger, column 7, lines 15-17 and column 8, lines 9-14). Applicants therefore submit that Bittenger discloses authenticating the client, not a message. Applicant can find no reference in Bittenger regarding verify that a message in authentic. Thus, applicants submit that Bittenger does not disclose an endpoint configured to receive a message and verify the message as authentic.

Further regarding claim 55, the Examiner states, "Leach discloses the creation of a data stack used to store the results of the operations locally on the server, then allows the client to map back to the stored results stack by providing an address which allows direct transfer between the client and the server greatly reducing the processing overhead." Applicants respectfully disagree with the Examiner's interpretation of Leach.

Leach teaches direct stack-to-stack transfer of parameters between two processes executing concurrently on a single machine. Rather than marshaling and unmarshaling parameters when invoking a remote procedure call in a second process, Leach teaches the use of a shared data stack and that a operating system kernel process copies the input and output parameters between the two processes individual stacks. (Leach, column 4, lines 32-43). Further, Leach teaches directly copying data from a server process to a client process through the use of a kernel that can access the address spaces of each process, thereby facility such a direct transfer to data (Leach, column 5, lines 44-52).

Thus, applicant submits that Leach fails to disclose the creation of a data stack used to store the results of the operations locally on the server, then allows the client to map back to the stored results stack by providing an address as the Examiner contends.

Specifically, under Leach, the client is not provided an address to map back to the any stored results stack, and the Examiner contends. Leach teaches that a kernel process directly transfers the output parameters onto the client's stack and "transfers control to the client process with the instruction pointer point to the instruction that follows the client process' call to the proxy method" (Leach, column 5, lines 52-55). Hence, Leach's kernel transfers any results directly into the client's stack and does not provide the client any address or information that would allow the client to "map back" to the stored results stack. Further, Leach teaches that the kernel must do this, because the client would not be able to access the other process' stack (Leach, column 6, lines 51-54).

In light of the above remarks, Applicants assert that the rejection of claim 55 is not supported by the cited art and withdrawal of the rejection is respectfully requested.

Applicants also assert that numerous ones of the other dependent claims recite further distinctions over the cited art. However, since the independent claims have been shown to be patentably distinct, a further discussion of the dependent claims is not necessary at this time.

Information Disclosure Statement:

Applicants note that three additional information disclosure statements with accompanying Forms PTO-1449 were submitted on August 17, 2001, September 17, 2001 and December 17, 2002, respectively. Applicants request the Examiner to carefully consider the listed references and return copies of the signed and initialed Forms PTO-1449 from each statement.

CONCLUSION

Applicants submit the application is in condition for allowance, and notice to that effect is respectfully requested.

If any extension of time (under 37 C.F.R. § 1.136) is necessary to prevent the above referenced application from becoming abandoned, Applicants hereby petition for such extension. If any fees are due, the Commissioner is authorized to charge said fees to Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C. Deposit Account No. 501505/5181-67300/RCK.

Also enclosed herewith are the following items:

☒ Return Receipt Postcard

☐ Petition for Extension of Time

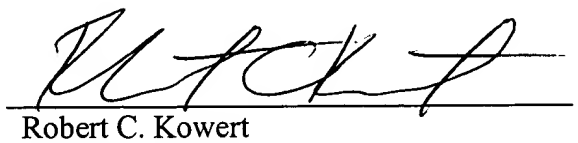
☒ Replacement Drawing Sheet

☐ Notice of Change of Address

☐ Fee Authorization Form authorizing a deposit account debit in the amount of $ for fees ( ).

☐ Other:

Respectfully submitted,



Robert C. Kowert
Reg. No. 39,255
ATTORNEY FOR APPLICANT(S)

Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C.
P.O. Box 398
Austin, TX 78767-0398
Phone: (512) 853-8850

Date: April 8, 2004